MAY/JUNE 1989 ISSUE # 24

VZ 200/300

HUNTER VALLEY VZ JOURNAL

HAPPY THIRD BIRTHDAY

PRODUCED BI-MONTHLY BY H.V.VZ.U.G.

APOLOGIES - The promised articles should appear in future issues.

COMING ISSUES :-

CUSTOMIZING E & F WORD PROCESSOR PRINTER MENU FOR TAPE USERS
ATARI TYPE JOYSTICK ADAPTOR FOR VZ 200/300 - EPROM ERASER
128K S/WAYS RAM FOR 4000H-5FFFH RANGE (DOS AREA)
AUTO START/STOP FOR DATASSETTES
34K USER RAM + 2K RAM (6000H RANGE) FOR VZ200

EDITORIAL - AS YOU MAY HAVE GATHERED THE JOURNAL IS CELEBRATING IT'S THIRD BIRTHDAY AND I'M VERY SURPRISED IT HAS LASTED SO LONG AND I CAN ONLY PUT DOWN IT'S SUCCESS TO IT'S CONTRIBUTORS BOTH REGULAR AND OCCASIONAL AND TO ALL THE SUBSCRIBERS WHO HAVE SUPPORTED US OVER THE YEARS.

SINCE MY CAR ACCIDENT LAST YEAR THERE WERE SEVERAL TIMES WHEN I NEARLY CALLED IT QUITS BECAUSE I HAD TOO MANY PROBLEMS PILE UP ALL AT ONCE WHICH KNOCKED ME FOR SIX. I FOUND THE HARD WAY I WASN'T SUPERMAN AFTER ALL. ON MAY 31 MY JOB BECOMES REDUNTANT AFTER 31 YEARS OF SERVICE AND I'LL BE AMONG THE UNEMPLOYED.

LOOKING ON THE BRIGHT SIDE AND NOT DRIVING TRUCKS ANYMORE SHOULD HELP RECOVERY PROCESS AND IT SHOULD GIVE ME MORE TIME TO SPEND ON THE VZ. I HEAR THE JOB'S OK, BUT THE PAY IS LOUSY. ALSO I'LD LIKE TO THANK YOU ONCE AGAIN FOR YOUR CONTINUED WISHES FOR MY SPEEDY RECOVERY, MUCH APPRECIATED.

THANKS - MY THANKS GO TO OUR REGULAR AND OCCASIONAL CONTRIBUTORS FOR THEIR CONTINUED SUPPORT OF THE HUNTER VALLEY VZ JOURNAL. CONTRIBUTORS LIKE DAVE MITCHELL, ROBERT QUINN, LARRY TAYLOR, BOB KITCH, BRIAN GREEVE AND MANY OTHERS HAVE MADE THE JOURNAL WHAT IT IS TODAY BY SHARING THEIR KNOWLEDGE AND EXPERTISE WITH OTHER VZ USERS. PLEASE KEEP IT UP AS WE NEED YOUR EXPERTISE.

MAY MEETING - ADAM MAGEE DEMONSTRATED HIS AMIGA 1000 AND EVERYONE PRESENT WAS IMPRESSED AND WE THANK ADAM FOR A VERY WELL PRESENTED DEMONSTRATION.

ELECTIONS - ON JUNE 2 AT OUR ANNUAL GENERAL MEETING A NEW COMMITTEE WAS ELECTED AND FOR A CHANGE WE HAD MORE NOMINEES THAN POSITIONS WITH EVERY MEMBER PRESENT AND ONE ABSENTEE MEMBER VOLUNTEERING. SIX NOMINATIONS WERE ACCEPTED WITH THE FOLLOWING MEMBERS BEING SUCCESFUL :-

ROSS WOODS, JOE LEON, COLIN BRIDGE, PETER JONES, ANDREW IRVINE & GARY BULLEY

ROSS WOODS AND MYSELF WILL ACT IN OUR PREVIOUS CAPACITY TILL OUR COMMITTEE MEETING ON JUNE 30 WHERE THE VARIOUS POSITIONS WILL BE DECIDED ON. BECAUSE OF THE SIZE OF THE COMMITTEE IT HAS BEEN PROPOSED TO HAVE THREE EXECUTIVE AND THREE COMMITTEE MEMBERS.

NEXT COMMITTEE MEETING - JOE LEON'S PLACE - 22 DRURY STREET WALLSEND - 7.00 PM

FUTURE MEETINGS - IT HAS BEEN PROPOSED TO HAVE CLUB MEETINGS MORE ORGANISED WITH A FORMAT BEING ADOPTED AND SOME FORM OF DEMONSTRATION PLANNED FOR EACH MEETING AND FUTURE EVENTS OR DEMONSTRATIONS BEING ADVERTISED IN THE JOURNAL SO MEMBERS CAN MAKE PLANS TO ATTEND. PRELIMENARY PLANS HAVE BEEN MADE FOR THE NEXT TWO MEETINGS WHICH ARE :-

JULY 7 SESSION ON PRINTERS —— AUGUST 4 - COMMODORE 64 DEMONSTRATION

WANTED TO BUY - 64K MEMORY RAM PACKS AND VZ 200 6K RAM BOARDS WITH OR WITHOUT 2K RAM CHIPS (NEEDED FOR FUTURE PROJECTS & EXPERIMENTS). IF YOU CAN HELP OUT PLEASE CONTACT :-

JOE LEON 22 DRURY STREET WALLSEND N.S.W. 2287 PHONE (049) 51 2756

DISCLAIMER - EVERY EFFORT IS MADE TO ENSURE THE ACCURACY OF INFORMATION CONTAINED WITHIN BE IT GENERAL, TECHNICAL, PROGRAMMING, ETC. NO RESPONSIBILITY CAN BE ACCEPTED BY H.V.VZ.U.G. OR AUTHOR AS A RESULT OF APPLYING SUCH INFORMATION.

HELP WANTED BY DAVE MITCHELL 4

* HELP * HELP * HELP * HELP * HELP * HELP * HELP * HELP * HELP *

I AM COMPILING ANOTHER LIST AND THIS TIME IT IS ALL THE READ ONLY MEMORY (ROM) ADDRESSES. IF YOU COULD HELP ME IN THIS REGARD I WOULD BE MOST GRATEFUL.

THE COMPLETED LIST WILL APPEAR IN A FUTURE JOURNAL AS A SUPPLEMENT. SO IF YOU KNOW OF ANY ADDRESSES SEND THEM TO THE SECRETARY/EDITOR OR TO ME DIRECTLY.

DAVE MITCHELL
24 ELPHINSTONE ST.
NORTH ROCKHAMPTON
QUEENSLANDD 4701

* HELP * HELP * HELP * HELP * HELP * HELP * HELP * HELP * HELP *

IC PIN OUTS FOR DOT MATRIX PRINTERS BY JEREMY LEE . . .

THIS PROGRAM IS DESIGNED TO LET YOU PRINT OUT PIN DESIGNATIONS ON KNOWN IC'S ON AN EPSON COMPATIBLE PRINTER. (SORRY, GP-100 OWNERS). THIS IS BECAUSE THE PROGRAM USES THE FULL 8 PINS ON THE EPSON PRINT-HEAD, AND THE GP-100 ONLY HAS SEVEN. THE PRINTER SHOULD BE EPSON COMPATIBLE BECAUSE THE PROGAM USES A LOT OF PRINT CODES TO GET THE GRAPHICS, AND HAS A WIDE PRINT HEADING, BUT ANYONE WHO KNOWS THE PRINT CODES FOR THEIR PRINTER SHOULD BE ABLE TO GET THE PROGRAM TO WORK WITH A LITTLE EFFORT.

WHAT'S THE PROGRAM FOR?, WELL, IF YOU'RE INTO ELECTRONICS, AND YOU'VE GOT ALL THE PIN DESIGNATIONS FOR A CHIP, BUT NO ACTUAL PRINT OUT, YOU USUALLY HAVE TO DRAW UP A LITTLE BOX WITH LINES STICKING OUT AND WRITE LITTLE THINGS NEXT TO IT. THIS IS OKAY, BUT IS MESSY. I SHOULD KNOW. IT ALSO TAKES A LONG TIME WHEN YOU USUALLY WANT TO GET ON WITH THE JOB.

THE SOLUTION, TYPE IN THIS PROGRAM. IT WILL PROBABLY JUST SIT ON SOME DUSTY, UNUSED TAPE/DISK FOR MOST OF IT'S LIFE, BUT IT CAN BE VERY USEFUL WHEN NEEDED. IT PRODUCES CLEAN LOOKING DIAGRAMS WITH "NOT" BARS AND NUMBERED PINS AND A NOTCH IN THE FRONT AN' EVERYTHING! (GREAT FOR DIAGRAMS IN MAGAZINES IF REDUCED A BIT ON THE PHOTOCOPIER).

MOST OF THE INSTRUCTIONS ARE INCLUDED IN THE PROGRAM ITSELF BUT THERE ARE A FEW THINGS YOU SHOULD KNOW.

1) THERE IS NO CORRECTION FACILITY, SO TAKE YOUR TIME. IF YOU TYPE SOMETHING WRONG AND HIT RETURN, YOU'LL HAVE TO BREAK THE PROGRAM AND START AGAIN. THIS ONLY GETS AGGRAVATING WITH REPEATED ERRORS OR IF YOU'RE DOING A 64 PIN IC AND YOU'RE NEAR THE END.

2) WHEN YOU SIGNIFY A "BAR" IT CAN ONLY GO OVER THE ENTIRE LINE, YOU CAN'T SAY SOMETHING LIKE " R/W " WITH BAR ONLY OVER THE "W".

THATS IT. A GOOD SUBJECT FOR THE FIRST TIME IS ONE OF THE VZ'S GATE ARRAY CHIPS. YOU'LL FIND THEM IN THE BACK OF THE TECHNICAL REFERENCE MANUAL. A FINAL NOTE, OF WHEN YOU TYPE IT IN AND THE PRINT SEEMS TO BE OVERLAPPING AS YOU LINE FEED, THEN CHANGE THE "CHR$(4)" IN LINE 305 TO "CHR$(8)".

```
0 CLEAR 5000
10 CLS
20 PRINT"          PIN-OUT PRINTER"
30 PRINT@64,"     HOW MANY PINS."
35 INPUTPN
36 IF(INT(PN/2)*2<>PN)OR PN<2 ORPN>70ELSE 40
37 PRINT@64,"  BE SERIOUS!       ":SOUND 10,8:GOTO 35
40 IFPN<20,S=1:GOTO 50
41 IFPN>28,S=2:GOTO 50
43 PRINT"  DOUBLE OR SINGLE WIDTH."
44 A$=INKEY$:IFA$<>"S"ANDA$<>"D",44
48 IFA$="D",S=2ELSES=1
50 L=INT(PN/2):DIMP$(PN):DIMB(PN):DIMPR$(L)
60 CLS
62 FORX=1TOPN
63 POKE 30777,1
64 PRINT@0,"                                      "
65 PRINT@32,"  ENTER PIN ASSIGNMENTS.          ";
66 PRINT"     ANTHING UP TO 8 CHARACTERS   WITH AN '*' AT THE END  TO
67 PRINT" SIGNIFY A BAR.                     ";
68 PRINT"--------------------------------":POKE 30777,35
70 PRINT@480,"PIN"X;:INPUTA$
75 IFRIGHT$(A$,1)="*",B(X)=1:A$=LEFT$(A$,LEN(A$)-1)
76 P$(X)=RIGHT$(A$,8)
80 NEXT
100 FORX=1TOL:A=LEN(P$(X)):P$(X)=LEFT$("         ",8-A)+P$(X)
101 NEXT
110 CLS:PRINT" WHAT'S THE TITLE FOR THIS       MASTERPIECE
120 INPUTT$
125 Q=6+S-INT(LEN(T$)/2):IFQ<1,Q=1
130 T$=LEFT$("                              ",Q)+T$
134 MW$="  "
135 INPUT"HOW MANY SPACES OVER";MW:FORX=1TOMW:MW$=MW$+" ":NEXT
140 LPRINTMW$;CHR$(14);T$
200 REM"SETUP CHARACTER CODES
205 GE$=CHR$(27)+CHR$(75)+CHR$(8)+CHR$(0)
210 S$=GE$:FORX=1TO8:S$=S$+CHR$(0):NEXT
220 B$=CHR$(27)+"K"+CHR$(6)+CHR$(0):FORX=1TO6:B$=B$+CHR$(6):NEXT
230 P$=CHR$(255):FORX=1TO8:P$=P$+CHR$(129):NEXT
240 E1$=CHR$(27)+"K"+CHR$(2)+CHR$(0)+CHR$(255)+CHR$(255)
250 P2$="":FORX=1TO7:P2$=P2$+CHR$(129):NEXT:P2$=P2$+CHR$(255)
260 N$=CHR$(224)+CHR$(24)+CHR$(8)+CHR$(4)+CHR$(2)+CHR$(2)
261 FORX=1TO4:N$=N$+CHR$(1):NEXT
265 N$=N$+CHR$(2)+CHR$(2)+CHR$(4)+CHR$(8)+CHR$(24)+CHR$(224)
270 E$="":FORX=1TO8:E$=E$+CHR$(3):NEXT
275 T$="":FORX=1TO8:T$=T$+CHR$(192):NEXT
300 REM"PRINT FIGURE
305 LPRINTCHR$(27);"A";CHR$(4):   '<--"CHANGE HERE IF NECESSARY
310 FORX=1TOL
315 AL=L+L-X+1
318 PR$=MW$
320 IFB(X)=0,PR$=PR$+"          ":GOTO340ELSEFORY=1TO LEN(P$(X))
330 IFMID$(P$(X),Y,1)<>" ",PR$=PR$+B$ELSEPR$=PR$+" "
335  NEXT
340 IFS=1,W=2:W2=6ELSEW=4:W2=10
341 PR$=PR$+" "+CHR$(27)+"K"+CHR$(2)+CHR$(0)+CHR$(0)+CHR$(0)
342  PR$=PR$+E1$:GOSUB 1000
350 IFX<>1,360
351 FORY=1TOW:PR$=PR$+GE$+T$:NEXT
352 PR$=PR$+CHR$(27)+"K"+CHR$(16)+CHR$(0)+N$
```

IC PIN OUTS FOR DOT MATRIX PRINTERS CONT. 6

```
353 FORY=1TOW:PR$=PR$+GE$+T$:NEXT
360 IFX<>1,FORY=1TOW2:PR$=PR$+S$:NEXT
370 GOSUB1000:PR$=""
371 PR$=PR$+E1$
378 GOSUB 1000
379 PR$=" "+CHR$(27)+"K"+CHR$(2)+CHR$(0)+CHR$(0)+CHR$(0)
380 IFB(AL)=0,400ELSEFORY=1TOLEN(P$(AL))
390 IFMID$(P$(AL),Y,1)<>" ",PR$=PR$+B$ELSEPR$=PR$+" "
395  NEXT
400 GOSUB  1000
410 PR$=""
415 LPRINT
416 PR$=MW$
418 N1$=MID$(STR$(X)+"  ",2,2)
419 N2$=RIGHT$(STR$(AL),2)
420 PR$=PR$+P$(X)+GE$+P$+E1$+N1$
430 FORY=1TOW2-3:PR$=PR$+S$:NEXT
440 PR$=PR$+N2$+E1$+GE$+P2$+P$(AL)
450 GOSUB 1000
455 LPRINT
460 NEXT X
463 PR$="          "+MW$
464 PR$=PR$+CHR$(27)+"K"+CHR$(2)+CHR$(0)+CHR$(0)+CHR$(0)
465 PR$=PR$+E1$:        FORY=1TOW*2+2:PR$=PR$+GE$+E$:NEXT
466 PR$=PR$+E1$:GOSUB 1000
470 LPRINTCHR$(27);"A";CHR$(10)
480 LPRINT" "
500 RUN
1000 IFPR$="",RETURN
1001 FORQ=1TOLEN(PR$)
1005 D=ASC(MID$(PR$,Q,1))
1010 IFINP(0)<>254,1010
1020 OUT 13,D:OUT 14,D
1030 NEXT:PR$=""
1040 OUT 13,0:OUT 14,0
1050 RETURN
```

HIDRAWER LLISTING ERROR . . . .

Those of you who typed in the LLISTing for HIDRAWER will have found that, when RUN, it BREAKed with a FUNCTION CODE ERROR message. The problem lies with the omission of the text of line 900 in the published LLISTing.

This arose because line 900 begins with a DEFINT statement. Most VZs are unable to display the DEFINT BASIC word when the DEFINT token is encountered, unless you have modified the WORD TABLE in ROM or, as I did, set up a modified WORD TABLE in battery backed RAM. Because the DEFINT statement was at the beginning of line 900, the text of the entire line was rendered invisible.

The following modifcation, spread over two BASIC lines, can be typed in to get the program running :-

```
900 FORA=30977TO30987:POKEA,2:NEXT
905 A=64:B=32:KK=3:CC=1:H=10:G=0:CK =3:K1=128
```

VECTORS & INTERRUPTS EXPLAINED BY BOB KITCH 7

IN VZ DOWN UNDER #15 A LETTER-TO-THE-EDITOR SOUGHT INFORMATION ON THE USE OF THE WORD "VECTOR" IN COMPUTER LITERATURE. OUR EDITOR PROVIDED A DICTIONARY DEFINITION OF THE TERM, HOWEVER A "DE-JARGONIZED" DEFINITION IS HEREIN PROVIDED. A PRACTICAL PROGRAM SHOWING HOW TO STEAL VECTORS TO MAKE AN INTERRUPT DRIVEN REAL TIME CLOCK ON THE VZ IS DETAILED.

DEFINITIONS.

THE WORD VECTOR HAS A NUMBER OF COMMON USEAGES - ALL ROUGHLY MEANING THE SAME - BUT HAVING SLIGHTLY DIFFERENT IMPLICATIONS ACCORDING TO THE PARTICULAR CONTEXT USED.

1. MY RECOLLECTION OF HIGH SCHOOL MATHS REMINDS ME THAT A "VECTOR QUANTITY" IS A PHYSICAL NUMBER HAVING A MAGNITUDE (OR SIZE) PLUS A DIRECTION, THEREBY DIFFERENTIATING IT FROM A "SCALAR QUANTITY" WHICH HAS SIZE ONLY. AN EXAMPLE OF EACH COULD BE VELOCITY (VECTOR) AND SPEED (SCALAR).

2. THE WORD VECTOR IS OFTEN USED IN NAVIGATION TERMINOLOGY FOR AIRCRAFT OR SHIPS. OFTEN A HEADING DIRECTION IS DESCRIBED AS "VECTOR 270" FOR EXAMPLE TO SUGGEST "GO WEST"!

3. THE DEFINITION PROVIDED BY HARRY IN VDU #15 IS A BIOLOGICAL ONE RELATING TO ORGANISMS THAT TRANSMIT GERMS OR DISEASE.

4. A LOOSELY RELATED NOTION, OFTEN USED IN DATA PROCESSING, DESCRIBES A VECTOR AS A NUMBER STORE OF FIXED LENGTH OR AS AN ORDERED LIST OF OBJECTS. A SINGLE DIMENSIONED ARRAY IN BASIC UTILIZES THIS CONCEPT AND MAY BE CALLED A VECTOR (EITHER ROW OR COLUMN) OR A LIST. A TWO DIMENSIONED ENTITY IS TERMED AN ARRAY OR TABLE AND HAS BOTH ROWS AND COLUMNS.

A THREE DIMENSIONED ENTITY MAY BE REFERRED TO AS A BLOCK AND HAS, IN ADDITION TO ROWS AND COLUMNS, LAYERS OR PLANES. THESE THREE FORMS OF DATA ARRAYS ARE ABOUT ALL THAT IS WORTH THINKING ABOUT - ALTHOUGH I HAVE WORKED WITH SCIENTIFIC PROGRAMS THAT HAVE EFFICIENTLY USED SEVEN DIMENSION ARRAYS! THE ADVANTAGE OF USING ARRAYS LIES IN THE ABILITY TO USE INDEXED ADDRESSING TO REFER TO A PARTICULAR CELL IN THE ARRAY..

5. ALL OF THESE USES OF VECTOR SUGGEST A MOVEMENT IN A SPECIFIC DIRECTION. MOREOVER, A STARTING CONDITION AND A FINAL CONDITION IS IMPLIED. IN COMPUTER JARGON THIS SUGGESTS A TYPE OF JUMP INSTRUCTION (OR IN BASIC - A GO TO INSTRUCTION). VECTOR HOWEVER DOES NOT REFER TO ANY-OLD-JUMP, AS A MICRO-PROCESSOR IS EXECUTING THIS TYPE OF INSTRUCTION OFTEN. NO, VECTOR IS RESERVED FOR A MORE POWERFUL FORM OF JUMP.

VECTOR AS A JUMP.

THE TECHNIQUE OF "VECTORING" GENERALLY REFERS TO THE ABILITY TO INTERCEPT ROM BASED ROUTINES AND CUSTOMIZE OTHER ROUTINES OR FACILITIES IN A MICROCOMPUTER. IN THIS SENSE, VECTORS ARE MERELY JUMPS TO MACHINE CODE ROUTINES. LET'S DIGRESS A LITTLE TO EXPLAIN THE ORGANIZATION OF THE ROM AND RAM IN THE VZ COMPUTER.

AS THE VZ EXECUTES A BASIC PROGRAM IT BRANCHES OUT OF ROM INTO THE RESERVED RAM (OR COMMUNICATIONS AREA) LOCATED FROM 7800H TO 7AE8H IN MEMORY. THE INTERPRETER DOES THIS FOR A NUMBER OF REASONS, ONE OF WHICH IS TO UPDATE THE SCREEN DISPLAY AND SCAN THE KEYBOARD. THIS HOUSEKEEPING HAPPENS 50 TIMES PER SECOND AND IS GENERATED BY AN INTERRUPT FROM THE VIDEO DISPLAY GENERATOR (VDG) CHIP. ANOTHER REASON IS TO PERMIT INTERCEPTION OF INSTRUCTIONS DURING PROGRAM EXECUTION. WITHOUT THIS FACILITY THE COMPUTER WOULD BE A FAIRLY INFLEXIBLE BEAST IF NO PROVISION WAS MADE FOR PROGRAM INTERCEPTION.

For example, if it was required to service the keyboard in a special way that hadn't ocurred to the ROM's authors, it would be necessary to implement a USR instruction to call a machine code routine. While this solution would work, it is like using a sledge hammer!

When the VZ initializes, it moves values from ROM into the Reserved RAM, thereby setting up a series of jump vectors for normal operation. Initialization occurs after power-up. Microsoft, the author of the Level II ROM, had enough foresight to provide vectors for most useful routines and checkpoints.

There is no doubt this was done for their own use, as later, a Disk BASIC could be easily added by stealing vectors (who said that a VZ could not be corrupted!). As the vectors reside in RAM they can be changed if desired. The behaviour of routines can be changed in a predictable fashion. Remember a vector is a jump in a prescribed direction.

The versatility and ease of use of vectors is the reason why their insertion, when writing ROM-based routines, is considered good programming practice. The point is that vectors were consciously put there to make this versatility available. Without them in a ROM-based system, one would be stuck with whatever is burned into the ROM. Fortunately for us, vectors are present in the VZ's BASIC ROM Interpreter.

The name given to this seemingly nefarious activity has a ring of felony about it - "vector stealing". It is a delightful description as I thoroughly enjoy every time I steal a vector and achieve an increase in the performance of the VZ!

As an example of the foregoing, I will briefly outline how a very useful "theft" can occur. When an LPRINT token is encountered in a BASIC line, the Interpreter vectors out of the ROM to location 79CAH which is located in the Reserved RAM area. Upon initialization this vector is set to C9H which is machine code for RETURN. The Z80 then vectors back into the ROM-based Interpreter and goes about printing out the following string of text. Herein lies the importance of the vector.

The printer routines burnt into the VZ's ROM are designed around the GP-100 type printers. These have largely fallen from favour for Epson-type printers now-a-days. By intercepting the three byte vector located at 79CAH to 79CCH it is possible to redirect the VZ into another routine adapted to the type of printer in use - this is a so-called "printer patch routine" loaded seperately into RAM. A three byte CALL or JUMP instruction can be written into the vector located at 79CAH to cause the redirection. Upon completion of printing the Interpreter returns to ROM again to resume execution and BASIC interpretation.

Pretty neat and useful eh!

Vector theft.

Often it is not wise to steal, but to borrow! Imagine that somewhere a keyboard input vector was stolen to facilitate some other form of input. By replacing the vector which "does a jump to keyboard" with a "jump to something else" then the keyboard scanning routine is bypassed.

Alternatively both routines could be scanned thereby providing an extra facility. A good example here is the use of the joysticks for input. Whilst the VZ routinely scans for keyboard input, it may be desirable to have the computer also scan for the joystick input ports. (See later)

Within the BASIC Interpreter installed in the VZ quite a few vectors are provided. Usually the vectors point straight back (RETURN) into the ROM routines. When a disk system is connected a number of these vectors are upgraded to provide an enhanced operating system. The problem is to find where these vectors are placed and what their use is. A disassembly of the ROM is required to discover what goes on inside the firmware.

The COPY-PROTECT utility developed by Larry Taylor intercepts a vector located at 79C1H to 79C3H and senses if the BREAK key has been pressed. If so, the BASIC program restarts by jumping to 1B5DH (RUN). This facilty disables listing, changing or copying a BASIC program. The borrowed vector causes a jump to a small machine code routine placed under the BASIC program. The vector at 79C1H is usually called from 032CH which is a routine in ROM used to output to either the screen or printer. This is a very smart use of vector stealing by Larry.

Another use of vector borrowing can be employed when the timing of events is required. As mentioned earlier in this article, the VZ vectors out to the "Interrupt Vector" every 20 msec. The Interrupt Vector is located at 787DH to 787FH and is usually called from 2EBCH or 3E37H in ROM. It is usually set to C9H or RETURN. The interrupt service routine commences at 2EB8H to 2EDAH. The routine at 3E37H to 3E3FH is called upon initialization to set the return vector at 787DH. By borrowing this vector various timing events can be set up.

Interrupts allow external events to get the immediate attention of the CPU. When some event happens, the interrupt rouses the CPU and forces it to carry out certain routines before resuming normal operation. It is a response to an external stimulus.

The Interrupt Vector.

Considerable discussion about the Interrupt Vector located at 787DH in RAM has been made. It is possible for the user to write system interrupt routines. The single interrupt generator is the VDG (as mentioned previously), which initiates an interrupt every 20 msec. for the Z80A CPU. This is introduced via the *INT pin on the Z80. This causes the Maskable Interrupt Mode in the CPU. Note that since the interrupt signal is the negative or down going edge of the vertical sync. pulse of the VDG, the Video RAM should only be accessed during the sync. pulse in the interrupt program.

The Video RAM should not be accessed at other times. The Z80 is set up to jump to 0038H upon receipt of the *INT line going negative. The interrupt service routine, resident in ROM is thereby entered. During the course of this latter routine control is passed out to the Interrupt Vector located from 787DH to 787FH. At this location the vector may be stolen. It is useful to provide a disassembled outline of this procedure as many users will wish to play around with this facility.

```
0038    JP 2EB8         ;To Interrupt Service Routine

2EB8    PUSH AF         ;Interrupt Service Routine
        PUSH BC
        PUSH DE
        PUSH HL
        CALL 787D       ;Interrupt Vector
        CALL SCREEN     ;Update screen
        CALL CURSOR     ;Flash cursor
        CALL KEYBRD     ;Check keyboard
        CALL BUZZER     ;Beep if keypressed
        POP HL
        POP DE
        POP BC
        POP AF
        EI                  ;ENABLE INTERRUPTS
        RETI                ;END OF INTERRUPT SERVICE
```

A RECENT PROGRAM PUBLISHED IN ETI USED THE TECHNIQUE OF BORROWING THE INTERRUPT VECTOR. A MODIFIED VERSION OF THE PROGRAM FOR A "REAL TIME CLOCK" ON THE VZ IS PROVIDED.

DISCUSSION OF REAL TIME CLOCK PROGRAM.

THE PROGRAM PROVIDED LOADS A SECTION OF MACHINE CODE AND ALLOCATES STORAGE FOR THE TIME-KEEPING PARAMETERS. FURTHERMORE, THE ROUTINE IS COMPLETELY RELOCATABLE IN MEMORY. THIS LATTER FEATURE ADDS SOME COMPLEXITY TO THE PROGRAM BUT TRY TO FOLLOW WHAT IS HAPPENING AS IT ACTUALLY MODIFIES THE MACHINE CODE ROUTINE AND IS AN EXTREMELY VERSATILE METHOD.

SUBROUTINES 30, 40 AND 50 ARE QUITE USEFUL AND GENERALIZED NUMERIC ROUTINES, EMPLOYED WHEN SIGNED INTEGERS (FOR DESCRIBING MEMORY LOCATIONS) ARE USED IN A LOADING ROUTINE.

TOP OF MEMORY IS LOWERED IN THE FAMILIAR MANNER IN LINES 210 TO 230 AND 99 BYTES OF MEMORY RESERVED. FOLLOWING THIS ACTION, ALL POINTERS ARE RESET AND VARIABLES DECLARED PRIOR TO DIMENSIONING. THIS SIGNICANTLY SPEEDS UP EXECUTION AS ARRAY VA% DOES NOT HAVE TO BE RELOCATED AS NEW VARIABLES ARE FOUND. INTEGER STORAGE IS USED THROUGHOUT TO ALSO SPEED UP EXECUTION.

THE MACHINE CODE, SET OUT IN LINES 3020 TO 3430, IS LOADED FROM LINES 310 TO 340. THE ADDITION OF A CHECKSUM VARIABLE ENSURES THAT INCORRECT DATA IS DETECTED. TRY AND FOLLOW THE "PSUEDO-ASSEMBLER" PROVIDED IN THE DATA STATEMENTS. IT IS A VERY SIMPLE PROGRAM.

LINES 410 TO 570 USE ARRAY VA% TO MODIFY THE ADDRESSES IN THE TIME KEEPING ROUTINE, THEREBY MAKING IT RELOCATABLE. THE DATA CONTAINED IN LINES 4010 TO 4300 ARE THE OFFSETS USED TO MODIFY THE ROUTINE. LINES 610 AND 620 POKE THE STARTING ADDRESS OF THE ROUTINE INTO THE INTERRUPT VECTOR.

LINE 630 IS "REMarked" OUT BUT IS A VALUABLE DEBUG DUMP OF THE PROCEDURE. ALSO THE DUMP CAN PROVIDE SOME INSIGHT INTO THE METHODOLOGY OF THE PROGRAM. THE CODE IN LINES 20020 TO 20120 SENDS THE DUMP TO THE PRINTER.

THE CURRENT TIME IS SET IN LINES 710 TO 750. RANGE CHECKING IS CARRIED OUT TO ENSURE PREDICTABLE PROGRAM EXECUTION.

FINALLY, THE INTERRUPT VECTOR IS "STOLEN" IN LINE 195 AND, 50 TIMES PER SECOND, THE ROUTINE IS ENTERED QUITE TRANSPARENTLY TO THE USER.

LINES 910 TO 980 ARE A LOOPING SECTION OF THE PROGRAM THAT CONTINUOUSLY DISPLAYS THE TIME AT THE BOTTOM OF THE SCREEN. THERE IS CONSIDERABLE SCOPE TO VARY THIS DISPLAY SECTION. LINE 950 IS "REMarked" OUT AND WILL PROVIDE A CLEAR SCREEN FOR THE TIME DISPLAY. ANOTHER OPTION IS SUGGESTED IN LINE 3420. WHENEVER THE TIME IS UPDATED, THE ROUTINE EXITS THROUGH THIS RETurn.

IF THE TIME IS NOT UPDATED, THE EXIT IS VIA THE CONDITIONAL RETurn IN LINE 3040. A SUITABLE SCREEN UPDATING DISPLAY DRIVER COULD BE APPENDED AFTER LINE 3430 THEREBY RENDERING THE BASIC DRIVER UN-NECESSARY. TRY AND DEVELOPE A SUITABLE DRIVER THAT WILL UPDATE THE SCREEN EVERY SECOND AND BE TRANSPARENT TO ANY OTHER USE OF THE SCREEN.

AS THIS PROGRAM WAS DEVELOPED ON A DISK SYSTEM, THE INCLUSION OF LINES 10000 AND 10010 ARE USED TO EASILY UPDATE THE DISK AS DEVELOPMENT PROCEEDS. SIMPLY ENTER RUN 10000 EVERY SO OFTEN TO SAVE THE LATEST VERSION.

PUTGET * 6 BY ROBERT QUINN 11

PUTGET*6 IS A BASIC ROUTINE THAT CAN BE ADDED TO A BASIC PROGRAM AND USED TO STORE AND RETRIEVE UP TO SIX COPIES OF THE 2K OF VIDEO MEMORY. THE ROUTINE COMES IN TWO PARTS :-

LINES 55100 TO 55130 ARE A SUBROUTINE THAT SETS UP A 12 BYTE MACHINE CODE PUT/GET ROUTINE STARTING AT 31348 IN THE COMMUNICATIONS REGION.

YOU WOULD PLACE A GOSUB55100 STATEMENT AT THE START OF ANY PROGRAM TO WHICH YOU HAVE ADDED PUTGET*6 SO THAT WHEN YOU RUN YOUR PROGRAM IT WILL SET UP THE M/C ROUTINE.

LINES 55000 TO 55090 ARE THE SUBROUTINE YOUR PROGRAM WOULD CALL WITH A GOSUB55000, ALLOWING YOU TO SELECT PUT (STORE VIDEO MEMORY) OR GET (RETRIEVE VIDEO MEMORY) AND TO SELECT WHICH ONE OF SIX STORES YOU PUT/GET TO/FROM.

HAVING GOSUB55000, PUTGET*6 IS IN PUT MODE. YOU CAN SWITCH TO GET MODE BY PRESSING NUMBER KEY <7> OR <8> OR <9>. YOU WILL HEAR A HIGH BEEP. IF YOU PRESS ONE OF THESE NUMBER KEYS AGAIN YOU WILL HEAR A LOW BEEP AND PUTGET*6 WILL BE SWITCHED BACK TO PUT MODE. EACH PRESS OF ANY ONE OF THESE THREE NUMBER KEYS SWITCHES BETWEEN PUT AND GET MODES, WITH THE IDENTIFYING BEEPS.

PRESS <Q> TO QUIT PUTGET*6 IF YOU CHANGE YOUR MIND.

THE NUMBER KEYS <1> TO <6> SELECT THE SIX STORES TO/FROM WHICH YOU PUT/GET COPIES OF VIDEO MEMORY.

THE FIRST STORE STARTS AT 49152; EACH SUBSEQUENT STORE IS 2048 (2K) MEMORY CELLS FURTHER ALONG THAN THE PREVIOUS. YOU WILL NEED A VZ300 + 16K MEMORY EXPANSION MODULE TO MAKE USE OF ALL SIX STORES.

STORE1: 49152 TO 51199	STORE2: 51200 TO 53247
STORE3: 53248 TO 55295	STORE4: 55296 TO 57343
STORE5: 57344 TO 59391	STORE6: 59392 TO 61439

ANY BASIC PROGRAM TO WHICH YOU ADD PUTGET*6 SHOULD, ALONG WITH ITS VARIABLE LIST TABLES WHEN RUN, NOT EXCEED 49152. IT IS GOOD PROGRAMMING TO INITIALISE ALL VARIABLES THAT WILL BE USED WHEN A PROGRAM IS RUN, IN A SUBROUTINE THAT IS GOSUBED AT START OF A PROGRAM. BY DECLARING ALL VARIABLES IN THIS WAY, THE LENGTHS OF THE VARIABLE LIST TABLES ARE FIXED AND YOU CAN FIND THE START OF FREE MEMORY AFTER RUNNING THE PROGRAM AND BREAKING WITH THIS COMMAND :-

```
PRINTPEEK(30973)+PEEK(30974)*256
```

YOU WILL HAVE 17687 MEMORY CELLS AVAILABLE FOR YOUR PROGRAM AND ITS VARIABLE LIST TABLES BEFORE YOU BEGIN TO OVERWRITE THE START OF THE FIRST VIDEO STORE. THERE WILL BE 1736 CELLS FREE FROM THE END OF THE LAST STORE (#6) TO TOP OF MEMORY IF YOU ARE USING A DISK DRIVE, MORE IF YOU ARE NOT. AMPLE MEMORY FOR STRING SPACE AND BASIC STACK.

OR YOU MAY CHOOSE TO PROTECT THE STORE MEMORY BY LOWERING TOP OF MEMORY TO BELOW 49152. POKE30897,255:POKE30898,191 WILL LOWER TOP OF MEMORY TO 49151. THE MEMORY UP TO 49151 WILL THEN HAVE TO ACCOMODATE YOUR PROGRAM, ITS VARIABLE LIST TABLES WHEN RUN AND STRING SPACE AND BASIC STACK.

PUTGET*6 IS VARIABLE-FREE; IT CAN BE USED IN ANY BASIC PROGRAM WITHOUT YOU HAVING TO WORRY IF IT WILL CHANGE THE VALUE OF SOME VARIABLE BEING USED BY YOUR PROGRAM, OR, BY INTRODUCING NEW VARIABLES, EXPANDING THE VARIABLE LIST TABLES.

BEFORE CALLING THE M/C PUT/GET ROUTINE WITH A USR STATEMENT, PUTGET*6 SAVES THE CURRENT ADDRESS STORED IN THE USR POINTER (30862/3) AND RESTORES THAT ADDRESS AFTER USING THE M/C PUT/GET ROUTINE. SO IF YOUR PROGRAM MAKES USE OF THE USR STATEMENT, YOU DO NOT HAVE TO ALTER THE USR POINTER BEFORE AND AFTER GOSUB55000.

THE USR CALL STATEMENT I USED IN PUTGET*6 (LINE 55080) IS A POKE STATEMENT--POKEUSR(3),0--TO AVOID HAVING TO USE A VARIABLE --E.G. X=USR(X)--OR A PRINTUSR(STATEMENT, WHICH WOULD CAUSE A REVERSION TO TEXT MODE IF YOUR PROGRAM WAS IN HI RES GRAPHIC MODE. THE POKEUSR(3),0 STATEMENT EXECUTES THE M/C PUT/GET ROUTINE, THEN POKES A ZERO BYTE TO ADDRESS 3.

BUT SINCE LOCATION 3 IS IN ROM, THE POKE DOES NOT WRITE. EVEN IF YOU HAD BATTERY BACKED RAM INSTALLED IN PLACE OF THE ROM OPERATING SYSTEM, IT WOULDN'T MAKE ANY DIFFERENCE SINCE LOCATION 3 OF THE OPERATING SYSTEM HAS A ZERO BYTE ANYWAY.

BY LOWERING THE START ADDRESS OF STORE1 TO 45056, AN EXTRA TWO STORES BECOME AVAILABLE, THOUGH OF COURSE DECREASING THE MEMORY AVAILABLE FOR BASIC PROGRAMMING. TO MODIFY PUTGET*6 TO MAKE IT PUTGET*8, CHANGE THESE STATEMENTS:-

```
IN LINE 55020, IFPEEK(31346)<9THEN55040
IN LINE 55050, POKE31350,176+8*(PEEK(31346)-1)
IN LINE 55060, POKE31353,176+8*(PEEK(31346)-1)
```

NUMBER KEYS <1> TO <8> WILL NOW SELECT STORES TO PUT/GET TO/FROM, AND ONLY NUMBER KEY <9> WILL SWITCH BETWEEN PUT AND GET MODES.

THE LOCATION OF THE MACHINE CODE PUT/GET ROUTINE HAS BEEN CHOSEN SO AS NOT TO CONFLICT WITH THE LOCATION OF PUTGET*3 (SEE ISSUE 19 OF JOURNAL). THIS MEANS YOU CAN USE PUTGET*3 TO PLACE THREE LO RES SCREENS IN THE VIDEO STORES, THEN USE PUTGET*6 TO STORE THOSE THREE STORED SCREENS PLUS THE CURRENT SCREEN IN ONE OF THE SIX PUTGET*6 STORES. ALTOGETHER YOU CAN STORE 24 LO RES SCREENS IN THE SIX PUTGET*6 STORES, AND RECALL THEM IN BATCHES OF FOUR, THEN USE PUTGET*3 TO VIEW THE THREE ALTERNATIVE SCREENS IN VIDEO MEMORY.

```
55000 POKE31347,0:SOUND20,2;30,1
55010 POKE31346,VAL(INKEY$):POKE31346,VAL(INKEY$)
55015 IFINKEY$="Q"THENSOUND9,2;20,1:RETURN
55020 IFPEEK(31346)=0THEN55010ELSEIFPEEK(31346)<7THEN55040
55025 IFPEEK(31347)=0THENSOUND30,1:POKE31347,1:GOTO55010
55030 SOUND20,1:POKE31347,0:GOTO55010
55040 SOUND9,1:IFPEEK(31347)=0THEN55060
55050 POKE31350,192+8*(PEEK(31346)-1):POKE31353,112:GOTO55070
55060 POKE31350,112:POKE31353,192+8*(PEEK(31346)-1)
55070 POKE31346,PEEK(30862):POKE31347,PEEK(30863)
55080 POKE30862,116:POKE30863,122:POKEUSR(3),0
55090 POKE30862,PEEK(31346):POKE30863,PEEK(31347):RETURN
55100 POKE31348,33:POKE31349,0:POKE31350,192:POKE31351,17
55110 POKE31352,0:POKE31353,112:POKE31354,1:POKE31355,0
55120 POKE31356,8:POKE31357,237:POKE31358,176:POKE31359,201
55130 RETURN
```

A DIRECTORY LABEL UTILITY BY DAVE MITCHELL . . . . .

SOME OF YOU WOULD HAVE SEEN THE LABELS THAT JOE HAS PLACED IN THE DIRECTORY. THESE LABELS WERE WRITTEN TO THE DIRECTORY BY A ROUTINE USED IN EXTENDED DOS. THE DIRECTORY CAN HOLD 120 FILENAMES, I HAVE NEVER SEEN ANYONE FILL THE DIRECTORY UP. SO I STARTED USING LABELS TO PLACE MARKERS FOR DATING ETC.

I HAVE WRITTEN A SMALL BASIC PROGRAM FOR YOU TO MAKE YOUR OWN LABELS, YOU CAN ADD TO IT OR INCLUDE IT IN YOUR PROGRAMS. TYPE THE FIRST SECTION AND SAVE IT BEFORE YOU RUN THE PROGRAM. THIS FIRST SECTION PLACES THE MACHINE CODE INTO LINE 10.

```
00010 REM...........................................................
00020 FORI=31470TO31523:READA:POKEI,A:B=B+A:NEXT
00030 IFB<>6065THENPRINT"ERROR IN DATA":END
00040 FORI=31524TO31530:POKEI,0:NEXT:LIST
00100 DATA229,243,205,8,64,14,50,6,255,4,205,56,64,219,19,183,62
00110 DATA4,250,14,64,58,33,121,183,32,18,205,47,64,183,194,14
00120 DATA64,34,25,123,205,11,64,251,225,201,201,254,205,50,64
00130 DATA183,194,14,64,24,239
```

AFTER YOU RUN THE PROGRAM, IT WILL LIST LINE 10 ONLY AND WILL READ AS BELOW IF YOU HAVE GOT AN EXTENDED BASIC IN MEMORY LINE 10 WILL READ A BIT MORE.

```
00010 REMPEEKLEN@2I39+8@INP
```

LEAVE LINE 10 IN MEMORY AND TYPE IN LINES 20 TO 150. AGAIN SAVE THE PROGRAM BEFORE YOU RUN.

LABELS MAY BE UP TO 10 CHARACTERS LONG AND YOU CAN PLACE UP TO 120 LABELS INTO THE DIRECTORY IF YOU WISH. BY TYPING DIR....... AS A LABEL THE DIRECTORY WILL BE PRINTED TO THE SCREEN.

```
00020 POKE30862,238:POKE30863,122
00030 CLS:PRINT"   THIS SMALL PROGRAM PLACES"
00040 PRINT"   LABELS INTO THE DIRECTORY"
00050 PRINT"  FOR EASE OF IDENTIFICATION."
00060 PRINT"   LABELS MAY BE UP TO 10"
00070 PRINT"   CHARACTERS LONG."
00080 PRINT@224,"ENTER LABEL   ..........":PRINT@236,;:INPUTA$
00090 IFA$="DIR......."THEN140
00100 A$=A$+"          "
00110 X=USR(0):B=PEEK(31513)+256*PEEK(31514)-65536
00120 FORI=0TO9:POKEB+I,ASC(MID$(A$,I+1,1)):NEXT
00130 FORI=10TO15:POKEB+I,0:NEXT:X=USR(1):GOTO80
00140 CLS:DIR:STATUS:PRINT"PRESS RETURN WHEN READY";
00150 A$=INKEY$:IFINKEY$=CHR$(13)THEN30ELSE150
```

SAMPLE DISK LABELS :-

```
PROGRAMS     FOR        JOURNAL    00 00 0000 0000 0000
FROM DAVE    MITCHELL   22:03:1989 00 00 0000 0000 0000
```

THE BASIC PROGRAM WAS COMPILED INTO THE WORD PROCESSOR FILE BY ANOTHER PROGRAM WHICH I CALLED "BSTWP.F". IT READS BASIC OR EDITOR ASSEMBLER SOURCE FILES FROM DISK AND MAKES WORD PROCESSOR FILES FROM THEM. THE BSTWP.F PROGRAM WILL BE INCLUDED WITH THE WORD PROCESSOR PATCH3.3 WHEN PURCHASED FROM THE CLUB. IF YOU ALREADY HAVE PATCH3.3 THEN CONTACT JOE LEON FOR YOUR COPY.

A CHECK DISK ROUTINE WRITTEN BY DAVE MITCHELL . . .

THIS IS ANOTHER OF MY SMALL ROUTINES THAT IS USEFULL IN IT'S OWN WAY. I USE CHECK DISK WHEN I GET AN INPUT/OUTPUT ERROR AFTER I HAVE INITIALIZED A DISK. THE PROGRAM READS EACH SECTOR OF THE DISK AND CHECKS FOR ERRORS. THIS PROGRAM DOES NOT STOP THE ERROR BUT IT WILL RETRIEVE SOME DISKS FROM THE SCRAP HEAP. THE PROGRAM WILL REDUCE THE STORAGE AREA ON THE DISK IF IT CANNOT FIND TRACKS OR SECTORS, THE AMOUNT OF FREE SPACE LEFT ON THE DISK (STATUS) WILL BE PRINTED TO THE SCREEN AT THE END OF THE PROGRAM.

WITH IBM & COMPATABLE COMPUTERS A CHECK IS MADE OF DISKS AFTER THEY ARE FORMATTED. IF SECTORS CANNOT BE FOUND, WHOLE TRACKS ARE LOCKED OFF FROM THE USER.

WITH THE VZ A CHECK IS ALSO MADE BUT WHEN ERRORS ARE FOUND IT PRINTS THE DREADED INPUT/OUTPUT ERROR.

CHECK DISK DOES NOT LOCK OFF WHOLE TRACKS JUST THE SECTORS THAT THE ERROR OCCURRED IN.

JOE ALSO SUGESTED "ANOTHER DISK Y/N" OPTION BUT I HAVE DISCARDED THIS AS IT IS MORE DATA TO TYPE AND CHANCES FOR ERROR. SO IF YOU WANT TO RUN THE PROGRAM AGAIN AND YOU HAVE NOT RUN ANY OTHER PROGRAM YOU CAN TYPE:-

```
POKE30862,PEEK(30884): POKE30863,PEEK(30885): PRINTUSR(0)
```

THIS WILL RETURN YOU TO THE START OF THE CHECK DISK PROGRAM.

CHECK DISK CAN ALSO BE USED ON A DISK THAT HAS PROGRAMS STORED ON IT AS CHKDSK PLACES THE TRACK MAP INTO MEMORY AND READS THE DISK THEN SAVES THE MAP TO DISK.

IF YOU SAVE A PROGRAM AND GET AN I/O ERROR THEN ERASE THE PROGRAM AND BRUN CHKDSK THIS SHOULD REPAIR THE ERROR.

AFTER THE CHKDSK PROGRAM HAS RUN A MESSAGE WILL APPEAR ON THE SCREEN PRESSING RETURN THE PROGRAM WILL CHECK THE DISK IN THE DRIVE THAT CHKDSK WAS LOADED FROM.

PRESSING THE 1 KEY WILL CHECK THE DISK IN DRIVE 1 AND PRESSING 2 KEY WILL CHECK THE DISK IN DRIVE 2.

WHEN THE 1 OR 2 KEYS ARE PRESSED THE DRIVE THAT CHKDSK WAS LOADED FROM WILL RESET TO TRACK ZERO BEFORE SELECTING THE RELEVANT DRIVE. AGAIN THIS IS TO TRY AND STOP THE BASHING OF THE DRIVE HEADS.

TAKE CARE WHEN TYPING THE LINES OF DATA :-

THE BASIC PROGRAM :-

```
00010 FORI=-28672TO-28057:READA:POKEI,A:B=B+A:NEXT
00020 IFB<>57691THENPRINT"ERROR IN DATA":END
00030 BSAVE"CHKDSK",9000,9268
00100 DATA205,201,1,33,139,145,205,117,43,205,73,0,254,13,40,36
00110 DATA254,50,40,8,254,49,32,241,62,16,24,2,62,128,245,243,205
00120 DATA8,64,253,126,20,183,40,4,71,205,62,64,205,11,64,241,253
00130 DATA119,11,253,54,17,0,253,54,18,0,243,205,80,52,205,8,64
00140 DATA1,50,0,205,56,64,205,53,64,183,40,6,33,214,145,195,35
```

```
00150 DATA145,253,52,17,62,15,253,190,17,32,234,205,53,64,183,194
00160 DATA205,144,253,54,17,0,253,54,18,1,253,110,49,253,102,50
00170 DATA253,94,52,253,86,53,1,80,0,237,176,243,205,53,64,183
00180 DATA194,44,145,253,52,17,62,16,253,190,17,32,238,253,54,17
00190 DATA0,253,52,18,62,40,253,190,18,32,224,253,110,49,253,102
00200 DATA50,229,54,0,209,213,19,1,127,0,237,176,209,253,110,52
00210 DATA253,102,53,1,80,0,237,176,205,91,145,253,54,17,15,253
00220 DATA54,18,0,205,50,64,183,40,6,33,4,146,195,35,145,253,110
00230 DATA49,253,102,50,17,0,0,14,78,6,8,126,203,15,56,1,19,16
00240 DATA249,35,13,32,242,213,213,225,205,175,15,33,50,146,205
00250 DATA117,43,225,229,203,60,203,29,203,60,203,29,203,60,203
00260 DATA29,205,175,15,62,46,205,58,3,225,62,7,165,60,71,33,131
00270 DATA255,17,125,0,25,16,253,205,175,15,33,67,146,205,117,43
00280 DATA205,11,64,195,25,26,253,110,52,253,102,53,253,126,18
00290 DATA61,203,39,95,22,0,253,126,17,254,8,63,237,90,230,7,60
00300 DATA71,78,203,1,203,9,16,252,203,193,71,203,9,203,1,16,252
00310 DATA113,195,135,144,219,19,203,127,200,205,11,64,251,33,76
00320 DATA146,205,117,43,33,185,145,205,117,43,58,175,122,183,32
00330 DATA250,205,73,0,254,13,32,249,243,205,80,52,205,8,64,1,50
00340 DATA0,205,56,64,201,31,32,32,32,32,32,32,32,32,32,67,72,69
00350 DATA67,75,32,68,73,83,75,13,32,32,32,32,87,82,73,84,84,69
00360 DATA78,32,66,89,32,68,46,77,73,84,67,72,69,76,76,13,32,32
00370 DATA32,87,72,69,78,32,82,69,65,68,89,32,80,82,69,83,83,32
00380 DATA82,69,84,85,82,78,13,0,69,82,82,79,82,32,73,78,32,68
00390 DATA73,82,69,67,84,79,82,89,32,83,69,67,84,79,82,83,32,84
00400 DATA82,89,32,32,82,69,70,79,82,77,65,84,84,73,78,71,13,0
00410 DATA69,82,82,79,82,32,73,78,32,83,84,65,84,85,83,32,83,69
00420 DATA67,84,79,82,32,84,82,89,32,32,32,32,32,32,82,69,70,79
00430 DATA82,77,65,84,84,73,78,71,13,0,32,83,69,67,84,79,82,83
00440 DATA32,70,82,69,69,32,32,32,0,75,32,70,82,69,69,13,13,0,13
00450 DATA82,69,77,79,86,69,32,87,82,73,84,69,32,80,82,79,84,69
00460 DATA67,84,32,76,65,66,69,76,0
```

BSTWP.F REVIEW BY JOE LEON

As you may have been aware I've been using Dave Mitchell's PATCH3.3 and previous patches together with E & W Word Processor for a long time to compile the Journal. Dave has never been one to rest on his laurels so it's no surprise to see continuing improvements to Patch 3.3.

```
00150 DATA145,253,52,17,62,15,253,190,17,32,234,205,53,64,183,194
00160 DATA205,144,253,54,17,0,253,54,18,1,253,110,49,253,102,50
```

The latest addition is a little routine that comes with Patch 3.3 which will convert your BASIC programs and source code files into Patch 3.3 Word Processor files. Using printer control codes and print menu makes it a lot easier to print out file in any mode or pitch your printer is capable off.

```
00004 ;05:02:89
00005        XOR   A
00006        LD    (7818H),A
00007        LD    A,60H
00008        LD    (7C3CH),A
```

The leading zeroes in the line numbers are a result of conversion of BASIC program or source code to word processor file. As you can see in this review that text, BASIC and source code files can be mixed in any order.

HARDWARE MODIFICATIONS, REVIEWS PLUS HINTS & TIPS

ISSUE	PAGE/S	DESCRIPTION
# 1	4-5	- VZ Inputs & Outputs by Joe Leon
# 1	10-11	- VZ Speech Synthesiser by Dave Boyce
# 2	9	- VZ 200 Super II Mods by Joe Leon
# 2	9-10	- Compumuse by Dave Boyce
# 3	6-8	- Printer Buffer by Dave Boyce
# 3	12-13	- VZ300 18K Memory Expansion by Dave Boyce
# 4	5-7	- 2K (6000H-67FFH) Battery Backed Ram by Dave Boyce
# 4	8	- Serial Interface - DSE Terminal
# 4	9	- Printer Buffer II by dave Boyce
# 4	15	- VZ Ritty Kit
# 5	5-7	- VZ 200 External Keyboard Mods by Dave Boyce
# 6	7-8	- Soft Start Interrupt Switch by DAVE MITCHELL
# 8/9	11-12	- Disk Write Protect Override by Joe Leon
# 8/9	16-17	- Shift Lock Switches by Joe Leon
#10/11	11-12	- The Magic VZ Eprom by Joe Leon
# 12	12	- 2K Zero Power Ram by Joe Leon
# 12	16	- Making Cartridge WP Work by Joe Leon
# 12	20	- AEM4505 Speech Synth. Mod by Dave Boyce
# 13	6-8	- 8K Built In Lithium Battery Ram I by Joe Leon
# 13	10	- VZ200/300 16K Ram Pack Mods by Ross Woods
# 13	15-16	- AEM 4505 Tone Control by Dave Boyce
# 14	13	- 32K Built In Lithium Battery Ram by Joe Leon
# 14	14-16	- 8K Built In Lithium Battery Ram II by Joe Leon
# 15	19	- VZ 200/300 Membrane K/Board by Joe Leon
# 16	5	- Improving VZ Sound Quality by Joe Leon
# 16	12-13	- 8K Built In Lithium Battery Ram III by Joe Leon
# 16	16	- VZ Membrane Keyboard Update by Joe Leon
# 17	9	- Fixing AC Plug Pack By Robert Gregg
# 18	6	- Shift Lock Update by Joe Leon
# 18	8-11	- 24 Bit I/O Interface by Bob Kitch
# 18	15	- NMI Switch Update by Joe Leon
# 18	16-17	- 128K Sideways Ram Part I by Joe Leon
# 19	6	- Tape Signal Conditioner by Neville Hughes
# 19	7	- Printer Buffer Update by Dave Boyce
# 19	8	- 64K Ram Pack Circuit by Chris Hobrough
# 19	9	- Hints & Tips on Dos Problems
# 20	8	- Printer Buffer Update III by Dave Boyce
# 20	9-10	- 32K Ram - 64K Eprom PCB'S from Dave Newcombe
# 20	14	- 4K-64K Ram/Eprom (6000H-67FFH) by Joe Leon
# 21	6	- Disk Drive Problems & Hints & Tips by Joe Leon
# 21	7	- Disk Drive Extension Lead by Joe Leon
# 21	14-15	- 128K Sideways Ram Part II by Joe Leon
# 21	16	- Head Cleaning Disk Update by Joe Leon
# 22	16-19	- VZ Super Graphics Part I by Joe Leon
# 23	14-15	- 128K Sideways Ram Part III by Joe Leon
# 23	16-18	- VZ Super Graphics Part II by Joe Leon

HARDWARE & SOFTWARE REVIEWS

ISSUE	PAGE/S	DESCRIPTION
# 1	6	- Games Review - Dawn Patrol
# 1	7	- Utility Review - Program Copier
# 2	7	- Games Review - Learjet
# 3	6-8	- Hardware Review - Printer Buffer by Dave Boyce
# 4	8	- Hardware Review - Serial Interface - DSE Terminal

HARDWARE & SOFTWARE REVIEWS CONTINUED . . . .

ISSUE	PAGE/S	DESCRIPTION
# 4	9	HARDWARE REVIEW - PRINTER BUFFER II BY DAVE BOYCE
# 4	15	HARDWARE REVIEW - VZ RITTY KIT
# 5	12	GAMES REVIEW - GALAXON
# 6	5-6	UTILITY REVIEW - HACKER'S DELIGHT
# 6	12-13	GAMES REVIEW - STAR BLASTER
# 8/9	7	UTILITY REVIEW - D. MITCHELL WP PATCH TAPE/DISK
#10/11	13	UTILITY REVIEW - LARRY TAYLOR FILESEARCH V2.0
# 12	6	UTILITY REVIEW - QUICKWRITE
# 12	13	UTILITY REVIEW - DAVE MITCHELL EXTENDED DOS V1.0
# 12	17	UTILITY REVIEW - DAVE MITCHELL WP PATCH 3.1
# 13	14	GAMES REVIEW - GALAXON
# 15	4-5	UTILITY REVIEW - LARRY TAYLOR PRINTER PATCH V1.4
# 15	14	MANUAL REVIEW - S. OLNEY ASSEMBLY MANUAL
# 17	4	UTILITY REVIEW - L. TAYLOR DISK MENU (B.VZ.U.W)
# 17	15	GAMES REVIEW - GALACTIC EMPIRES
# 18	7	UTILITY REVIEW - R. QUINN TAPE/DISK DATA BASE
# 19	5	GAMES REVIEW - VZ MONOPOLY
# 19	7	HARDWARE REVIEW - PRINTER BUFFER UPDATE
# 20	8	HARDWARE REVIEW - PRINTER BUFFER UPDATE

ARTICLES

ISSUE	PAGE/S	DESCRIPTION
# 1	8-9	VIDEO DISK? - BINARY FILE COMMANDS - R. QUINN
# 3	10-11	MODE(1) PEEK & POKE - MATTHEW TAYLOR
# 3	14-16	UNDERSTANDING YOUR VZ PART I - R. QUINN
# 4	10-12	UNDERSTANDING YOUR VZ PART II - R. QUINN
# 5	8-9	PEEK VS INKEY$ - R. QUINN
# 5	11	UNDERSTANDING YOUR VZ PART III - R. QUINN
# 6	9-11	INTRODUCTION TO PROGRAMMING PART I - R. KITCH
# 6	14-18	UNDERSTANDING YOUR VZ PART IV - R. QUINN
# 7	4	PRINT/LPRINT SWITCH - R. QUINN
# 7	5-6	PROGRAMMING FOR BEGINNERS PART I - L. ROGERS
# 7	9-10	UNDERSTANDING YOUR VZ PART V - R. QUINN
# 8/9	8-10	INTRODUCTION TO PROGRAMMING PART II - R. KITCH
# 8/9	18-20	UNDERSTANDING YOUR VZ PART VI - R. QUINN
#10/11	7-8	RAM COMMUNICATION ADDRESSES - D. MITCHELL
#10/11	9-10	ROM INVERSE CHARACTER SET - R. KITCH
#10/11	15-16	PROGRAMMING FOR BEGINNERS PART II - L. ROGERS
#10/11	17-19	ENHANCING VZ BASIC PART I - L. TAYLOR
# 12	4	ADDITIONS TO RAM COMM. REGION - R. QUINN
# 12	7	"BOOLEAN LOGIC OPERATORS" - R. KITCH
# 12	17-19	ENHANCING VZ BASIC PART II - L. TAYLOR
# 12	14-15	USING DISK TOKENS PART I - R. QUINN
# 12	18-19	ENHANCING VZ BASIC PART II - L. TAYLOR
# 13	11-13	VZ TOKENS & WORDS PART I - R. QUINN
# 13	17-18	DOT MATRIX PRINTERS PART I - L. TAYLOR
# 14	5-7	CHARACTER CODES - R. QUINN
# 14	11-13	USING DISK TOKENS PART II - R. QUINN
# 14	17-18	VZ TOKENS & WORDS PART II - R. QUINN
# 15	17-19	ENHANCING FIND PART I - L. TAYLOR
# 16	6-7	ENHANCING FIND PART II - L. TAYLOR
# 17	9	ENHANCING FIND PART III - R. QUINN
# 17	10-11	DOT MATRIX PRINTERS PART II - L. TAYLOR
# 17	12-13	D.M.P. GRAPHICS EDITOR - L. TAYLOR
# 18	13-15	BANK SWITCHING 64K RAM PACK - C. HOBROUGH
# 22	4-5	THE VZ - WHERE TO FROM HERE - L. TAYLOR
# 23	4-5	NUMBER SYSTEMS - L. TAYLOR
# 24	7-10	VECTORS & INTERRUPTS EXPLAINED - R. KITCH

PROGRAMS & PROGRAMMING . . . .

ISSUE	PAGE/S	DESCRIPTION
# 1	9	- Hi-Res Screen Dump For D.M.P.
# 1	12	- Word Search - Dave Boyce
# 2	5-6	- A Challange - Two Basic Games
# 2	11	- Hints For Beginners - Useful POKEs
# 3	5	- Play With This "Meteor Game" - Jamie Perry
# 3	17-19	- Hints For Beginners
# 3	20	- Hints For Dos Users
# 4	13-14	- For Beginners - Maths Teacher
# 4	16	- Hints for beginners
# 5	9-10	- Printer/Plotter Section - Dave Boyce
# 5	13	- For Beginners - Hi-Res Drawing
# 6	4	- Disk Menu Maker I - Joe Leon
# 7	6	- Printer/Plotter Hints & Tips
# 7	7-8	- Disk Menu Maker II - Joe Leon
# 8/9	4-6	- Snoopy Calendar (DMP) Part I - Dave Boyce
# 8/)	7	- Merging E&F W.P. Tape/Disk Files - Joe Leon
#10/11	4-5	- Snoopy Calendar Part II - Dave Boyce
#10/11	5&10	- Disk Drive Hints & Tips
#10/11	6	- Snoopy Calendar For Printer/Plotter - Dave Boyce
#10/11	14	- DSE Tech. Bulletin #111 - Mailing List To Disk
#10/11	20	- Mode(0) Lo-Res Graph Worksheet
# 12	6	- Rejuvinating Printer Ribbons
# 12	9	- Improving S.O. Extended Basic V2.3 - D. Mitchell
# 12	10-11	- Hints For Beginners - Andrew Laird
# 12	11	- Sideways Print Routine For D.M.P. - Dave Boyce ✓
# 13	3	- Programming Hints & Tips
# 13	4-6	- Hi-Res Drawing Routines - Don Isles
# 13	9-10	- Base Converter - Dave Mitchell
# 14	3	- Disk Drive Primer - Hints & Tips
# 14	4	- Elements - Educational - Paul Leon
# 14	7	- Mailing List (Disk) Update II - Joe Leon
# 14	8-10	- Speech Synthesiser Programs - Dave Boyce
# 14	10	- INKEY$ INPUT Routine - Paul Leon ✓
# 15	5-6	- Mental Maths - Educational - John Garland
# 15	7	- Kaleidoscope - Hi-Res Pattern Maker - R. Quinn ✓
# 15	8-9	- Capturing CSAVE/CLOAD - Dave Mitchell
# 15	10-13	- Mailing List (Disk) Update III - Joe Leon
# 15	15-16	- Basic Block Transfers Part I - Chris Hobrough
# 16	4	- Spelling Checker Aid - Educational - Paul Leon
# 16	8-9	- Basic Block Transfers Part II - Chris Hobrough
# 16	10-11	- Mailing List (Disk) Update IV - Joe Leon
# 16	14-16	- Lprinter For Printer/Plotter - Robert Quinn ✓
# 16	17-18	- Block Move & Compare - Dave Boyce ✓
# 17	5-6	- NSW Footy Tab Selector - Paul Leon ✓
# 17	7	- Hex Constants - Dave Mitchell ✓
# 17	8	- Car Running Expenses - Joe Leon ✓
# 17	14	- Using Head Cleaning Disk - Joe Leon
# 17	16-18	- Interrupt Function Keys - Robert Quinn ✓
# 18	4-6	- Maths Tester - Educational - Brooke Spong
# 18	11-12	- Jagular Jetty I - Adam Magee ✓
# 19	4	- PutGet*3 - Robert Quinn ✓
# 19	9	- Dos Problems
# 19	10-11	- Jagular Jetty II - Adam Magee ✓
# 20	4-5	- Pages Disk Menu - Paul & Joe Leon
# 20	6-7	- Suite - Robert Quinn
# 20	11	- 60 Digit Multiply Routine - Neville Hughes ✓
# 20	12-13	- Beam Heading - Des Holmes ✓
# 20	15	- Musical Joysticks - B. Gregg ✓

CONTINUED IN NEXT ISSUE . . . .

CONTRIBUTIONS TO THE HUNTER VALLEY VZ JOURNAL :-

IF YOU ARE THINKING OF CONTRIBUTING TO THE JOURNAL THE PREFERED FORMAT IS BASIC LISTINGS, WORD PROCESSOR OR SOURCE CODE FILES ON TAPE OR DISK. FILES FROM THE FOLLOWING WORD PROCESSORS CAN BE ACCEPTED :-

E & F TAPE OR DISK, WORDPRO CARTRIDGE AND ALL QUICKWRITE WPORDPROCESSOR FILES.

WANTED TO BUY - 64K RAM PACKS - CONTACT JOE LEON
22 DRURY STREET WALLSEND NSW 2287 --- PHONE (049) 51 2756

MEETINGS - FIRST FRIDAY OF MONTH - NO MEETING IN JANUARY 1990

VENUE - JESMOND NEIGHBOURHOOD CENTRE MORDUE PARADE JESMOND
(REAR STOCKLAND MALL - BIG W)

JUNE 2 ANNUAL GENERAL MEETING - ELECTION OF NEW COMMITTEE
JULY 7 - AUGUST 4 - SEPTEMBER 1 - OCTOBER 6 - NOVEMBER 3 - DECEMBER 3

IF YOU ATTEND ONLY ONCE A YEAR THEN PLEASE COME TO THE JUNE MEETING AS THE CLUBS FUTURE IS IN YOUR HANDS. IT CANNOT FUNCTION WITHOUT YOUR SUPPORT.

HUNTER VALLEY VZ USERS' GROUP - P.O. BOX 161 JESMOND N.S.W. 2299

SECRETARY/EDITOR-JOE LEON (049) 51 2756
PRESIDENT-ROSS WOODS (049) 71 2843

TREASURER - MATTHEW TAYLOR ---- COMMITTEE MEMBER - COLIN BRIDGE

SUBSCRIPTION - H.V.VZ.JOURNAL - 6 MONTHS $9.00 - 12 MONTHS $18.00
NEW ZEALAND - 6 MONTHS $12.00 - 12 MONTHS $24.00

VZ USER - NO WORD AS YET IF ANYONE IS TAKING OVER ITS PUBLICATON

LE'VZ OOP - VZ MAGAZINE - VSOFTWAREZ/SOFTWARE/HARDWARE FOR SALE
J.C.E. D'ALTON 39 AGNES ST. TOOWONG QUEENSLAND 4066

VZ DOWN UNDER - VZ MAGAZINE - 6 ISSUES - $18.00 PER YEAR
HARRY HUGGINS 12 THOMAS ST. MITCHAM 3132 VICTORIA

VZ-LINK - PETER J. HILL P.O.BOX 1972 C.P.O. AUCKLAND N.Z.

WAVZ - GRAEME BYWATER P.O. BOX 388, MORLEY W.A. 6062

BRISBANE VZ USERS WORKSHOP - C/O 63 TINGALPA ST. WYNUM WEST 4178
SOFTWARE FOR SALE - DISK MENU

NOTE :- WHEN WRITING TO ANY ABOVE OR H.V.VZ. USERS' GROUP FOR INFORMATION PLEASE ENCLOSE A S.S.A.E. OR NZ 2 INT. REPLY COUPONS.

FOR SALE - E & F WORD PROCESSOR PATCH 3.3 . . . .

PATCH 3.3 WRITTEN BY DAVE MITCHELL WILL CONVERT YOUR E & F TAPE WORD PROCESSOR FOR FULL DISK USE WHILE RETAINING ALL ORIGINAL FUNCTIONS. BELOW ARE THE DISK COMMANDS & ADDED FUNCTIONS :-

LOAD, SAVE, ERASE, RENAME, DIRECTORY, INITIALIZE, UPDATE, DRIVE 1 & 2, SHIFTLOCK & IMBEDDED PRINTER CONTROL CODES PLUS CTRL+P WHICH BYPASSES PRINT MENU AND PRINTS TO SCREEN OR PRINTER. A ROUTINE IS ALSO PROVIDED TO CONVERT YOUR BASIC PROGRAM OR SOURCE CODE FILES INTO WORD PROCESSOR FILES.

PATCH 3.3 HAS PROVISION FOR IMBEDDING PRINTER CONTROL CODES IN TEXT AND FAST SAVING AND LOADING OF TEXT DATA TO AND FROM DISK USING BLOCK SAVE/LOAD TECHNIQUES. PRINTER CONTROL CODES CAN BE SAVED TO TAPE OR DISK.

MINIMUM MEMORY REQUIREMENTS - VZ 300 + 16K RAM PACK - VZ 200 + 26K

PATCH 3.3 IS COPYRIGHT TO AND ONLY AVAILABLE FROM :-
HUNTER VALLEY VZ USERS' GROUP P.O.BOX 161 JESMOND 2299
N.S.W. AUSTRALIA - PHONE JOE LEON (049) 51 2756

PRICE - AUS/NZ AU$20.00 - UPDATE - AUS-$10.00 - NZ-AU$11.00.
UPDATING AVAILABLE ONLY TO PREVIOUS PURCHASERS OF OUR PATCHES.

FOR MORE INFORMATION WRITE TO H.V.VZ.U.G. ENCLOSING A SSAE.

EXTENDED DOS V1.3 - $15.00 - UPDATED VERSION WITH EXTRA COMMANDS ADDED.

OLD COMMANDS - MERGE, DIRA, LDIRA, DIRB, LDIRB, OLD, OLD., DEC, HEX, STATUSA AND LSTATUSA. STATUSA AND LSTATUSA ALSO WORKS WITH VERSION 1.0 DOS.

NEW COMMANDS :—
MENU - LOADS AND RUNS BINARY OR TEXT MENU PROGRAM FROM DISK.
CODE - SIMPLIFIES USING PRINTER CONTROL CODES DIRECTLY OR FROM WITHIN A PROGRAM.
LTAB - IS FOR SETTING OF LEFT MARGIN.
MOVE - MOVES BASIC FILE FROM DISK TO CHOSEN MEMORY ADDRESS.
UPD - ERASES OLD FILE AND SAVES WITH SAME FILE NAME.

MENU/FILE COPIER/DISK ORGANIZER - $15.00

THIS UTILITY WILL READ YOUR DISK DIRECTORY AND PRESENT YOU WITH SEVERAL OPTIONS. USING THE CURSOR YOU CAN RUN/BRUN ANY PROGRAM OR SELECT FILE COPY, REN, ERASE, DRIVE 1 OR 2, ETC. BESIDES COPYING TEXT AND BINARY FILES ALL OTHER FILES CAN BE COPIED AS WELL EXEPT FOR DATA FILES.

FOR PURCHASE OR INFORMATION CONTACT - DAVE MITCHELL - (079) 27 8519
24 ELPHINSTONE ST. NORTH ROCKHAMPTON QUEENSLAND 4701

FOR INFORMATION OR DEMONSTRATION IN NEWCASTLE AREA CONTACT :-
JOE LEON - (049) 51 2756 - 22 DRURY ST. WALLSEND NSW 2287

VZ SCREEN ED - $25.00 + $2.00 POST & HANDLING..

VZ SCREEN ED IS A NEW DRAWING PROGRAM FOR HI-RES SCREENS WITH MANY POWERFUL OPTIONS. BOTH TAPE AND DISK USERS ARE CATERED FOR.

SEND YOUR CHEQUE OR MONEY ORDER TO - OR PHONE MATTHEW ON (049) 75 2350
MATTHEW TAYLOR SOFTWARE 38 FISHING POINT ROAD RATHMINES NSW 2283

SEE MATTHEW AT CLUB MEETINGS FOR A DEMONSTRATION OR PURCHASE OF SCREEN ED